Integrated Circuit ORB (ICO) Whitepaper



www.adlinktech.com
2017

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Table of Contents

1. Synopsis .... P 3

2. Overview .... P 3

3. Detailed Description .... P 7

5. Conclusion .... P 8

Notices .... P 9

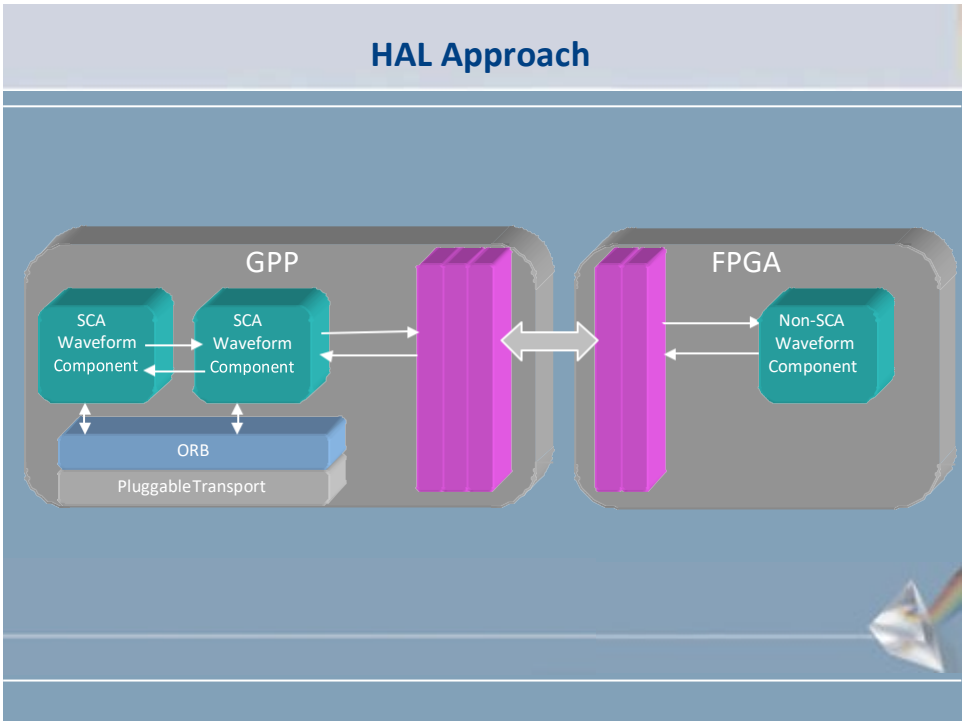
LRi Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

1. Synopsis

This white paper presents a detailed overview of ADLINK's Integrated Circuit ORB (ICO) and describes the product features and design environment. The ICO supports a drop-in Software Communications Architecture (SCA) compatible interface between distributed software objects running on processors and waveform objects residing in silicon. The connection between Software client and Hardware servant will now be seamless, fast and use fewer system resources.

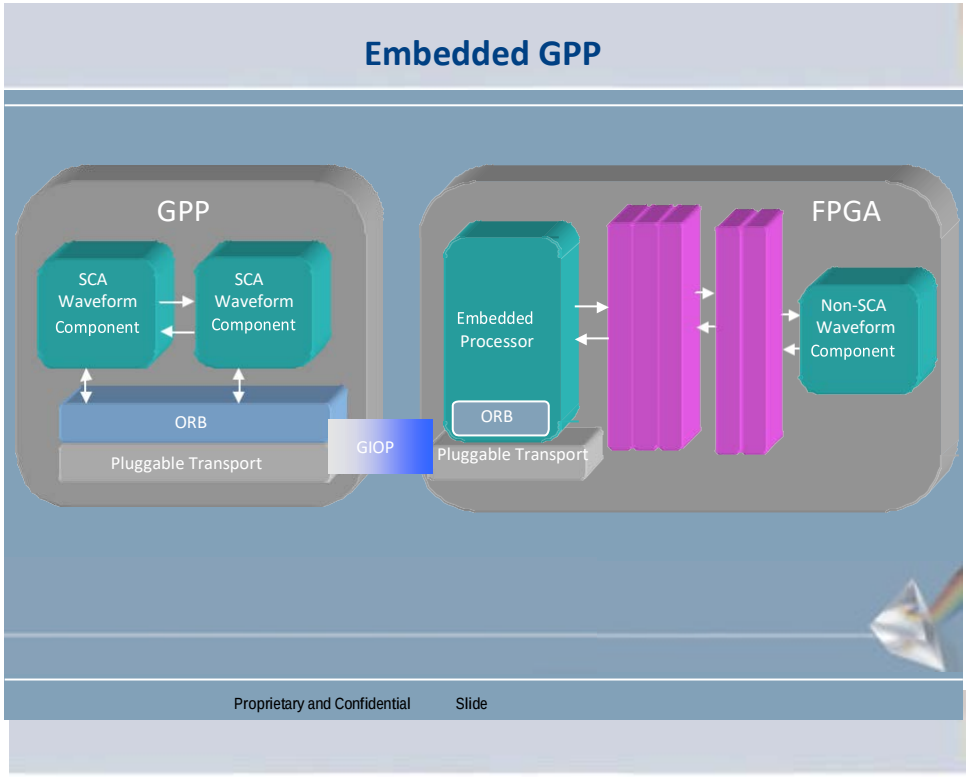
2. Overview

Using ICO, one eliminates the need to develop custom proxies on General Purpose Processors (GPPs) and Digital Signal Processors (DSPs) that simply serve to establish communication to waveform objects residing within FPGAs (Field Programmable Gate Arrays). These proxies (sometimes referred to as Hardware Abstraction Layers – HALs) are used when designing to Software Defined Radio (SDR) architectures such as the SCA and are meant to increase portability and re- use, but in practice, they tend to increase latency, reduce throughput, and lower re- use.



Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

ICO further eliminates the need to embed general purpose processing cores into FPGAs in order to offer software ORB capability. Although a viable approach, this approach tends to require significant gate count and memory utilization and generally these processing cores cannot be clocked fast enough to deal with the ever -increasing performance requirements of SDR applications.

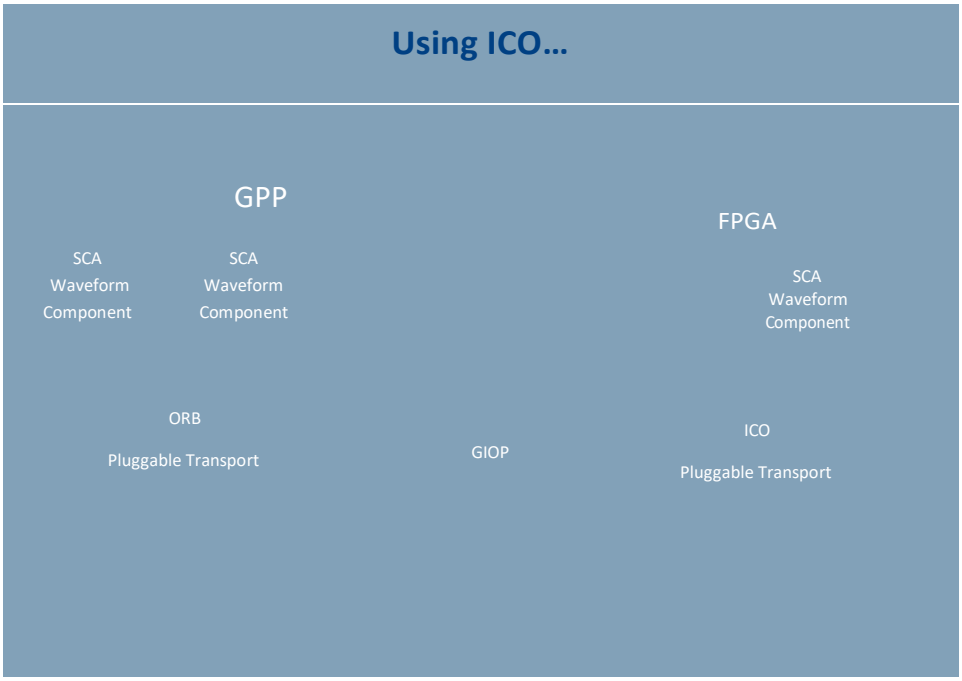


The embedded ORB has been written in portable VHDL that can be synthesized onto any FPGA or ASIC platform.

The ICO design environment consists of:

- The ICO engine,
- IDL to VHDL code generator,
- Spectra Modeling Tool,
- The SCA waveform component.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000



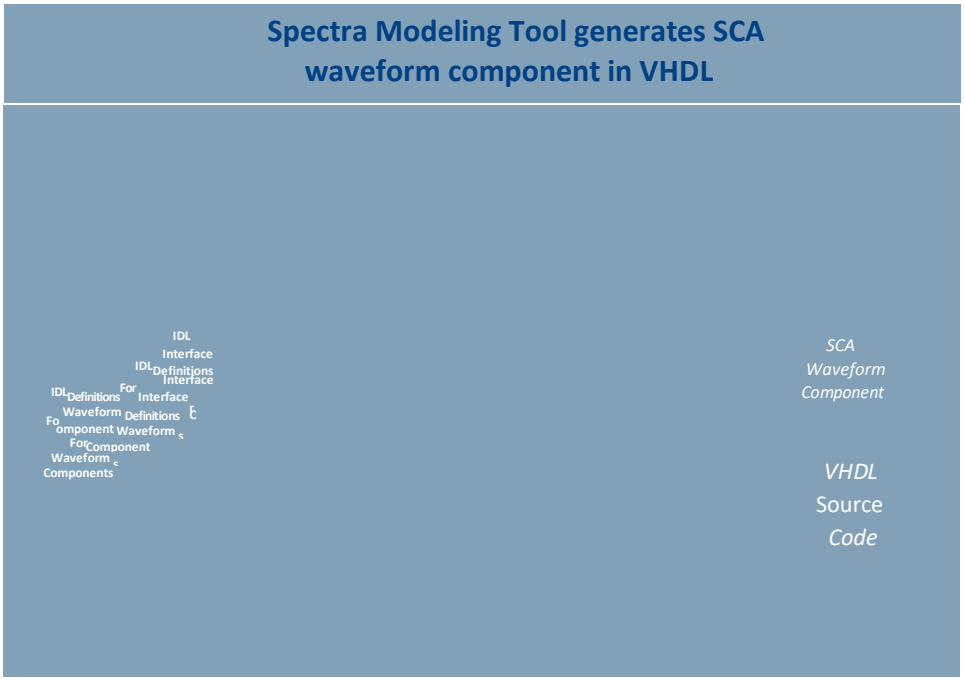
The ICO engine is responsible for implementing the transfer syntax used in CORBA messages. The engine unmarshals the incoming GIOP octet stream and extracts header and data fields while discarding padding. Endian conversion is performed on all incoming data based on information in the GIOP message header. In the incoming direction, the engine performs operation name demultiplexing to determine which object the data in the GIOP message is being transferred to. Message data is then extracted for transfer to the appropriate logic.

If a message indicates that a response is expected, the ICO engine generates a reply message. The engine will perform a read operation to an object, if necessary, to obtain data for the reply. It then populates the header field and aligns the data. When a reply message has been built, the ICO engine transfers the data to the outside world via a FIFO- like interface.

The IDL to VHDL code generator is part of ADLINK's IDL compiler family. This software tool is responsible for generating configuration parameters needed by the ICO engine to do operation name demultiplexing and data routing to the appropriate objects. The code generator also adds parameters to VHDL package files that configure the physical aspects of the ICO interface and internal storage elements. Parts of the VHDL code for the core are also generated at compile time by the code generator.

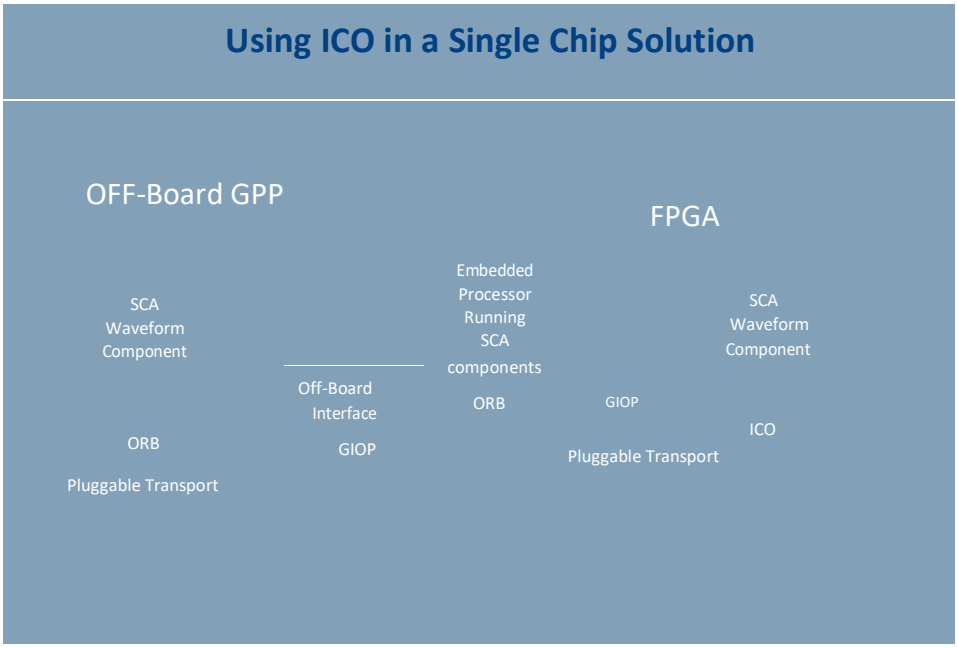
In an SCA-compliant environment, ICO communicates with the hardware developer's native waveform logic via an SCA waveform component. The VHDL for this component is generated by ADLINK's Spectra Modeling Tool. The ADLINK tools then stitch together the ICO and the SCA waveform component to present the developer with a single core.

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000



The hardware developer treats the ICO as any other IP interface core. The core can be instantiated in the HDL capture of the FPGA design between the native waveform logic and the system bus. The system side of the core appears as a typical FIFO interface. The native side of the core has a simple and open interface to communicate with the waveform logic.

Software developers treat ICO components as they would any other CORBA object. This design approach makes communication between the S/W and H/W objects seamless. Using ICO, radio developers can now host radio elements in an FPGA and still have them be addressable and callable from an SCA-compliant software core framework as though it was an SCA object and not an FPGA.



LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

3. Detailed Description

The ICO core resides between the waveform logic and the local transport and takes the place of the address decode block found in typical bus interface designs. The basic design process of the FGPA is unchanged as it relates to waveform and system bus performance considerations.

In typical designs, the local processor communicates with thewaveform objects inside the FPGA via the processor bus or some other local transport such as Ethernet. Each accessible register and memory within the FPGA is assigned a location in the processor's address map. Data bound for the FPGA is partitioned into packets with the address of the destination and may be further encapsulated in the format of the local transport. Logic on the receive side strips away the transport encapsulation and passes the data packets to the local address decode logic. The address is decoded and the data is sent to its location via a data bus along with the required read or write strobe.

In the case of ICO, each accessible register and memory is given an operation name and its I/O properties are described in Interface Description language (IDL). Registers can be accessed alone or in groups depending on how they are described in the IDL. If it is desired to write a register and read it back at a different time, the register would require two operation names; one for write and one for read. If it is desirable to write a register and read the results immediately it could be described as a single operation with an inout parameter. ADLINK provides tools that can aide hardware engineers in writing the IDL descriptions.

Once the registers and memories are described in terms of IDL and the compiler has parameterized ICO, a software ORB running on the local processor can access the registers via CORBA GIOP messages. The ORB encapsulates the data in a GIOP message and passes this to the local transport via the Extensible Transport Framework (ETF). The GIOP message may itself be encapsulated in the format of the local transport. In the case of Ethernet, each ICO might be assigned a unique MAC address in the local system. User designed logic on the receive side strips away the transport encapsulation and routes the GIOP messages to ICO. Depending on the size of the packet, the user may have to provide an intermediate storage such as a FIFO because ICO only has a small buffer memory on the system input side. The ICO interface on the system input side is designed to connect to a FIFO and contains signals such as data available, data _read, data_valid and data in.

ICO processes the GIOP message as described in the overview section above and sends handshake signals to the user waveform logic. In the case of a write, the user receives data and a write strobe. Each operation receives its own unique read and or write strobe depending upon how it was described in the IDL. The IDL to VHDL compiler produces a list of operation names and the strobes assigned to them so that the user can make the appropriate connections in the top level VHDL code. ADLINK tools can automatically make these connections for the user.

The write bus is shared by all waveform objects. Reads require that each waveform object have a unique read bus. The read busses will all be brought back to ICO where a wire-or will be done to multiplex them into a single bus. For this reason is it required that the user waveform logic drive its read bus low when not passing data on it. Further, it is probable that one waveform object will contain several operations depending on the user's implementation of the design. The user will therefore have to inform the compiler as to how many read busses are required so that ICO can be parameterized correctly. The width of the read a nd write busses will automatically be set by the compiler to the smallest data width specified in the IDL. All read and write signals are asserted for one clock cycle.

ICO asserts the read strobe and waits for an acknowledge back from the user waveform logic to know that the read data is valid. The data is then encapsulated in a GIOP message and stored in a memory internal to ICO. The size of this memory must be configured by the user via a compiler setting so that it can fit the maximum sized return data packet.

The system output interface of ICO looks like a FIFO. It contains signals such as data_available, data_valid, fifo_rd and data_out. Data available will not be asserted until an entire GIOP message is ready for transfer. User designed logic can then read the GIOP message from ICO and route it to the local transport for transmission back to the processor.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

4. Conclusion

ADLINK has created a powerful solution for implementing real-time CORBA in an SCA environment. The ICO is flexible, highly configurable and uses a minimum of FPGA resources. It frees the hardware developer from learning and implementing the complexities of CORBA protocols and allows concentration on custom waveform design elements. The software engineer is presented with a seamless environment in which to communicate between client and server applications. System developers have a solution that is portable across platforms sharing the same interconnect fabric. The ADLINK ICO represents a significant leap forward in SCA based tools and products.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Web: www.adlinktech.com

Email: ist_info@adlinktech.com

Notices

© 2017 ADLINK. All rights reserved. This document may be reproduced in whole but not in part. The information contained in this document is subject to change without notice and is made available in good faith without liability on the part of ADLINK Corporation. All trademarks acknowledged.

LRI Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000



Leading EDGE COMPUTING

All products and company name listed are trademarks or trade names of their respective companies.
Updated Sep. 13, 2016. ©2016 ADLINK Technology, Inc. All Rights Reserved. All specifications are subject to change without further notice.